IN THE CLAIMS

1. (Currently Amended) A method for creating a dynamic client side service proxy framework using meta-data and introspection capabilities of Open Grid Services Architecture (OGSA) service data, the method comprising:

defining an Open Grid Service Invocation Factory configured to create a service proxy;

introspecting an Open Grid Service Infrastructure (OGSI) service based on information exposed by the service; and

creating an OGSI Service Invocation Proxy defining a set of dynamic interfaces based on said service introspection and a meta-data inspection interface of said Service Invocation Proxy;

wherein said Service Invocation Proxy exposes both static port type interfaces and dynamic interfaces to support more flexibility of the client; and

wherein said OGSI service based information includes service data elements, port types, Grid Service Reference (GSR) values, and operation extensibility parameters.

2. (Cancelled)

3. (Original) The method of claim 1, wherein said dynamic interface includes a minimum of a Grid Service port type.

4. (Original) The method of claim 1, wherein said Service Invocation Proxy provides inspection features on the service including at least one of: service implemented port types, static and dynamic service data types, Qnames, and language specific types.

5. (Original) The method of claim 1, wherein said static port type interfaces include at least one of a port type and service interface.

6. (Original) The method of claim 1, wherein said dynamic interfaces include a common set of programming patterns.

7. (Currently Amended) The method of claim 1 ~~2~~, further comprising:

binding choices based on said GSR values, the GSR values including binding encoding information that is hidden from the client.

8. (Currently Amended) The method of claim 7, wherein said ~~GSR~~ binding encoding information includes Web Services Description Language (WSDL), WSIF, JAX-RPC and CORBA IOR..

9. (Currently Amended) The method of claim 8, wherein said binding encoding information is configured to support multiple transport binding information including SOAP/HTTP and SOAP/JMS.

10. (Original) The method of claim 1, further comprising;

refreshing said Service Invocation Proxy based on GSR lifetime information.

11. (Original) The method of claim 1, wherein said framework further comprises creation of a service data language types from extensible markup language (XML) schema types at runtime.

12. (Original) The method of claim 1, wherein said framework further comprises a pass through interface mechanism for web service call properties including security, transaction, logging and other information.

13. (Original) The method of claim 1, wherein said framework further comprises a caching mechanism for service types and GSR framework.

14. (Original) The method of claim 13, wherein said caching mechanism optimizes performance by avoiding round-trips to the service.

15. (Original) The method of claim 1, wherein said framework further comprises an introspection mechanism on service calls configured to support common programming "aspects" as defined by Aspect Oriented Programming concepts.

16. (Currently Amended) A system for creating a dynamic client side service proxy framework using meta-data and introspection capabilities of Open Grid Services Architecture (OGSA) service data comprising:

a grid client;

a defined Open Grid Service Invocation Factory configured to create a service proxy;

an Open Grid Service Infrastructure (OGSI) service in communication with said grid client via a communications network, said OGSI service includes OGSI service based information exposed by the service and introspected by said Factory; and

an OGSI Service Invocation Proxy defining a set of dynamic interfaces based on said service introspection and a meta-data inspection interface of said Service Invocation Proxy;

wherein said Service Invocation Proxy exposes both static port type interfaces and dynamic interfaces to support more flexibility of the client; and

wherein said OGSI service based information includes service data elements, port types, Grid Service Reference (GSR) values, and operation extensibility parameters.

17. (Cancelled)

18. (Original) The system of claim 16, wherein said dynamic interface includes a minimum of a Grid Service port type.

19. (Original) The system of claim 16, wherein said Service Invocation Proxy provides inspection features on the service including at least on of: service implemented port types, static and dynamic service data types, Qnames, and language specific types.

20. (Original) The system of claim 16, wherein said static port type interfaces include at least one of a port type and service interface.

21. (Original) The system of claim 16, wherein said dynamic interfaces include a common set of programming patterns.

22. (Currently Amended) The system of claim 16 ~~17~~, further comprising:

binding choices based on said GSR values, the GSR values including binding encoding information that is hidden from the grid client.

23. (Currently Amended) The system of claim 22, wherein said ~~GSR~~ binding encoding information includes Web Services Description Language (WSDL), WSIF, JAX-RPC and CORBA IOR.

24. (Currently Amended) The system of claim 23, wherein said binding encoding information is configured to support multiple transport binding information including SOAP/HTTP and SOAP/JMS.

25. (Currently Amended) The system of claim 23, further comprising;

refreshing said Service Invocation Proxy based on the GSR values. ~~lifetime information.~~

26. (Original) The system of claim 16, wherein said framework further comprises creation of a service data language types from extensible markup language (XML) schema types at runtime.

27. (Original) The system of claim 16, wherein said framework further comprises a pass through interface mechanism for web service call properties including security, transaction, logging, and other information.

28. (Original) The system of claim 16, wherein said framework further comprises a caching mechanism for service types and GSR framework.

29. (Original) The system of claim 28, wherein said caching mechanism optimizes performance by avoiding round-trips to the service.

30. (Original) The system of claim 16, wherein said framework further comprises an introspection mechanism on service calls configured to support common programming

"aspects" as defined by Aspect Oriented Programming concepts.